

Lab Assignment 4**Due 2030****Lab Grading Policy: Attendance 20%, Score 80%**

In case you have difficulty in finishing the exercises on time, you should upload them before **noon on Saturday** and a penalty of 20% discount will be applied on your score. No late submission is permitted. We will in general post the reference solutions **by Sunday**.

Download FirstMidterm_dist.zip **from the course website.**

Exercise 1 (30%): Rational numbers (fractions) are numbers that can be written in the form a/b , where a and b are integers and $b \neq 0$. a is known as the *numerator* and b the *denominator*. Let the default value for a be 0 and for b be 1. Implement a class `Fraction` to represent rational numbers and allow their objects to support the following client code (`Prob2_Client.cpp`):

```
#include <iostream>
#include "Fraction.h"

using namespace std;

int main(){
    Fraction f1; // 0/1
    f1.setName("f1");
    cout << "=====" << endl;
    printFraction(cout, f1);
    Fraction f2(3); //3/1
    f2.setName('f');
    Fraction f3(-2, 4); //-2/4
    cout << "=====" << endl;
    printFraction(printFraction(cout, f2), f3);
    cout << "=====" << endl;
    Fraction f4(cin); // prompt for input
    cout << "=====" << endl;
    printFraction(cout, f4);
    cout << "=====" << endl;
    printFraction(cout, f4.setName(&f2));
    return 0;
}
```

Below is a sample run:

```

=====
Fraction f1: 0/1
=====
Fraction f: 3/1
Fraction anonymous: -2/4
=====
Enter the name for Fraction: F4
Enter the values for numerator and denominator: 5 3
=====
Fraction F4: 5/3
=====
Fraction f: 5/3
請按任意鍵繼續 . . .

```

Exercise 2 (50%) (a) (30%) Write a `Matrix` class that reads its contents with double prescribed from input. We will use MATLAB-like syntax to parse the input. For example, if you have a matrix $A = \begin{bmatrix} 1.1 & 3.0 & 6.5 \\ 7.8 & 4.5 & 2.2 \end{bmatrix}$, the MATLAB-like input syntax discards white space and uses semi-colon to separate different rows and comma to separate the elements in a row. The syntax goes like:

```
A=[1.1, 3.0, 6.5; 7.8, 4.5, 2.2]
```

Notice that the number of elements in each row must be the same and your program should be smart enough to detect this error. You should also implement matrix addition operation and check for dimension consistency before performing the operation. Use the following client code (`Prob3a_Client.cpp`) and consult the sample run to test your implementation.

```

#include <iostream>
#include "Matrix.h"
using namespace std;

int main(){
    // m1 2x2
    cout << "Enter the contents of matrix m1 with MATLAB syntax" << endl;
    Matrix m1(cin);
    cout << "The contents of matrix m1 are: " << endl;
    printMatrix(cout, m1);
    cout << endl;
    // m2 inconsistent dimension
    cout << "Enter the contents of matrix m2 with MATLAB syntax" << endl;
    Matrix m2(cin);
    cout << "The contents of matrix m2 are: " << endl;
    printMatrix(cout, m2);
    cout << endl;
    // m3 2x4
    cout << "Enter the contents of matrix m3 with MATLAB syntax" << endl;
    Matrix m3(cin);
    cout << "The contents of matrix m3 are: " << endl;
    printMatrix(cout, m3);
    cout << endl;
}

```

```

    // m4 2x2
    cout << "Enter the contents of matrix m4 with MATLAB syntax" << endl;
    Matrix m4(cin);
    cout << "The contents of matrix m4 are: " << endl;
    printMatrix(cout, m4);
    cout << endl;
    // Addition
    cout << "The contents of m1 + m3 are: " << endl;
    printMatrix(cout, addMatrix(m1, m3));
    cout << "The contents of m1 + m4 are: " << endl;
    printMatrix(cout, addMatrix(m1, m4));
    return 0;
}

```

Below is a sample run:

```

Enter the contents of matrix m1 with MATLAB syntax
m1=[1.1, 2.2; 3.3, 4.4]
The contents of matrix m1 are:
1.1    2.2
3.3    4.4

Enter the contents of matrix m2 with MATLAB syntax
m2=[1.0, 3.5, 3.2; 1.3, 2.6]
The contents of matrix m2 are:
1      3.5    3.2
1.3    2.6
WARNING: dimensions are not consistent!!!

Enter the contents of matrix m3 with MATLAB syntax
m3=[1.5, 2.6; 1.3, 2.1; 1.4, 5.6]
The contents of matrix m3 are:
1.5    2.6
1.3    2.1
1.4    5.6

Enter the contents of matrix m4 with MATLAB syntax
m4=[2.2, 4.5; 1.35, 4.3]
The contents of matrix m4 are:
2.2    4.5
1.35   4.3

The contents of m1 + m3 are:
MATRIX ADDITION WARNING: two matrix dimensions are not consistent!
The contents of m1 + m4 are:
3.3    6.7
4.65   8.7
請按任意鍵繼續 . . .

```

(b) (20%) Relieve the input syntax and allow for more flexible parsing. Use the following client code (Prob3b_Client.cpp) and consult the sample run to test your implementation.

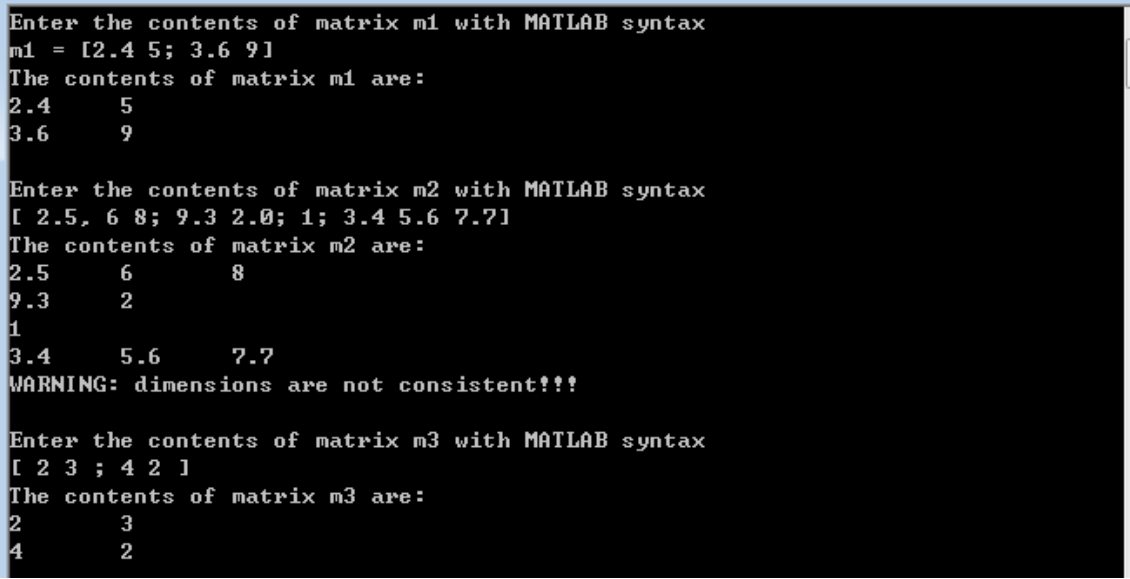
```

#include <iostream>
#include "Matrix.h"
using namespace std;

int main(){
    // m1
    cout << "Enter the contents of matrix m1 with MATLAB syntax" << endl;
    Matrix m1(cin);
}

```

```
    cout << "The contents of matrix m1 are: " << endl;
    printMatrix(cout, m1);
    cout << endl;
    // m2
    cout << "Enter the contents of matrix m2 with MATLAB syntax" << endl;
    Matrix m2(cin);
    cout << "The contents of matrix m2 are: " << endl;
    printMatrix(cout, m2);
    cout << endl;
    // m3
    cout << "Enter the contents of matrix m3 with MATLAB syntax" << endl;
    Matrix m3(cin);
    cout << "The contents of matrix m3 are: " << endl;
    printMatrix(cout, m3);
    cout << endl;
    return 0;
}
```



```
Enter the contents of matrix m1 with MATLAB syntax
m1 = [2.4 5; 3.6 9]
The contents of matrix m1 are:
2.4      5
3.6      9

Enter the contents of matrix m2 with MATLAB syntax
[ 2.5, 6 8; 9.3 2.0; 1; 3.4 5.6 7.7]
The contents of matrix m2 are:
2.5      6      8
9.3      2
1
3.4      5.6      7.7
WARNING: dimensions are not consistent!!!

Enter the contents of matrix m3 with MATLAB syntax
[ 2 3 ; 4 2 ]
The contents of matrix m3 are:
2      3
4      2
```